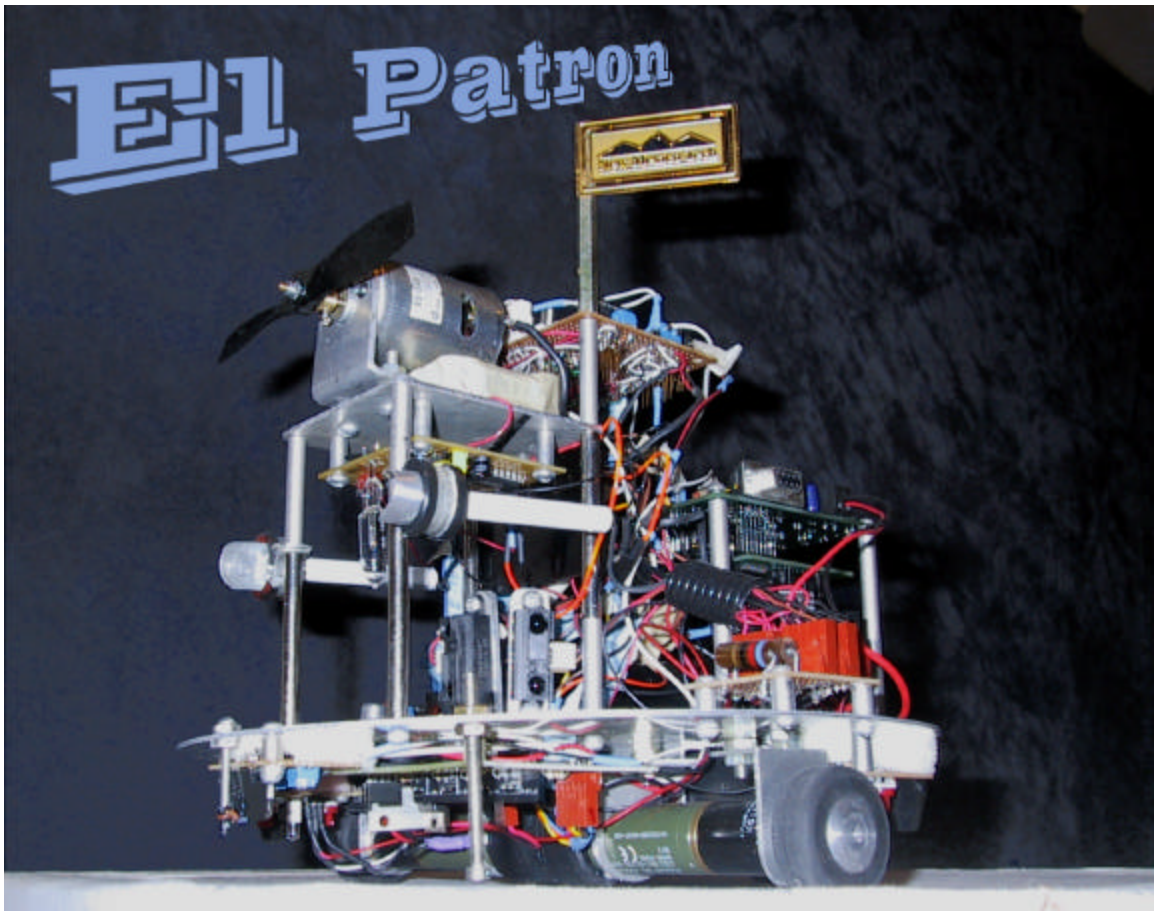



# EE 382 – Junior Design
# Firefighting Robot Project

Anthony Montoya, Jr.
Ivan Olguin
Randy Sena

May 7, 2001

# Table of Contents

# List of Figures

## Abstract

At the start of the spring semester of 2001, the junior design class was given a task to complete by the May 1, 2001. This task was to design and build a mobile robot capable of competing in the Trinity College Fire Fighting Home Robot Contest or other mobile robot projects approved by the instructors. The robot had to navigate through a maze, detect a flame and extinguish the flame using various sensors. The Motorola 68HC12 processing board was chosen for the brains of the robot. Each subsystem had to be interfaced using the HC12. The main goal of this project was to develop teaming skills, develop oral and written communication skills, and develop the ability to decompose large problems.

Each of the different subsystems that were involved in completing the above task are described in detail. We will also explain all the hard work and dedication that was involved in completing our goal. All of our hard work was rewarded because we were chosen to be one of the three groups to be sent to the National Trinity College Fire-fighting contest. We placed 21$^{st}$ in the nationals out of seventy-one entries and we placed 3$^{rd}$ at out local competition. The following technical report presents the results of our design for our robot El Patron.


## Introduction

The design of El Patron included various sub systems, which had to be integrated for communication with the Motorola HC 12. The robot's subsystems and circuitry include: white line sensors, wall sensors, fire sensors, and fire extinguishing, along with maze navigation and a return home routine. With all hardware systems integrated, the

processor can then communicate with all systems in effort to complete maze navigation and fire extinguishing. As a group of three, the work was split among us based on our strong points, which related to the robots design and construction. The following is a detailed report that includes descriptions and schematics relating to each system and operation of El Patron. The group goal was to build a compact, dependable robot that could consistently execute the fire maze navigation.

## Chassis Setup

The chassis set up of El Patron is clean and low profile. The center of gravity is very low due to the fact that the heaviest objects were mounted low to the ground. The heaviest objects are the motors and 12 volt battery. The motors are mounted very low because the tires are very small. Their wheel size is a small 1.5-inch diameter. By using a 2 motor differential drive system the motors had to be mounted on the center of the plate. Through the use of one caster wheel the robot was able to maintain stable balance. To prevent the robot from tipping forward, the 12-volt battery was mounted underneath the plate and toward the back. To keep the top of the plate clean, we were able to mount the board with the optical isolators, 5-volt regulator, and H Bridge all underneath the plate. With the need for three white line sensors we also had to mount them under the plate. We placed one in the front, and the other two on each side near the motor wheels.

With the underside of the plate filled, we then mounted more hardware on the top. The HC 12 was elevated with spacers and mounted directly on top of the 12-volt battery. Under the HC 12 we placed our 7.50-volt cell phone battery. On the front of the robot we mounted the fan on top of an elevated plate. Between the fan and plate we located two 9-

volt batteries. Underneath the fan plate was the Hamamatsu circuit and sensor. Directly under the Hamamatsu circuit was the placement of the three wall sensors. Mounted to the top of the fan plate was another circuit board. This board contained the frequency to voltage converters, single voltage Maxim 494 op amp, and A to D switching multiplexer. With only one small, elevated plate we were able to keep the chassis down to one main plate. By placing as much as we could underneath the plate we were able to keep the top of the robot looking clean. All hardware components were securely mounted and we had no trouble with sensors or circuits coming loose. The result of a good chassis set up allowed El Patron to have a clean appearance along with great reliability.

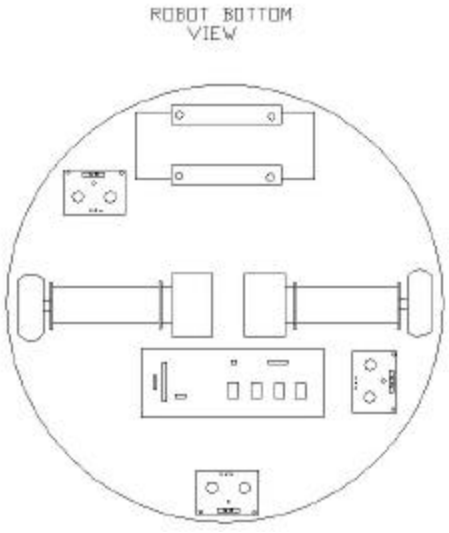



Figure #1

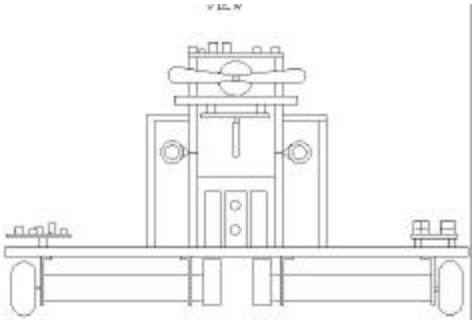


Figure #2

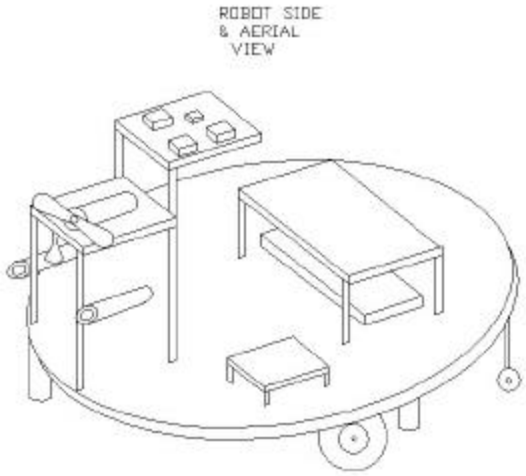



Figure #3

# Subsystems

## Power Systems

El Patron's power system consisted of 3 supplies. We used a 12V sealed lead acid battery to power the motors, motor drivers, fan, and regulator for the optical isolation circuit. The sealed lead acid was large and able to supply 2000mAh of current. This is what we considered to be our high power supply. The second battery source was a 7.2V Lithium-ion cell phone battery. This powered the Motorola, mc7805ct 5-volt regulator. The regulator then powered the HC-12 microprocessor, motor encoders, Sharp

wall sensors, white-line sensor circuits, tone decoder circuit, and the photo diode circuitry. Our third power supply was two 9V batteries in series. The 18V supply powered the Hamamatsu circuit board, the frequency to voltage circuit, and the Maxim 333 analog switch.

Power Distribution

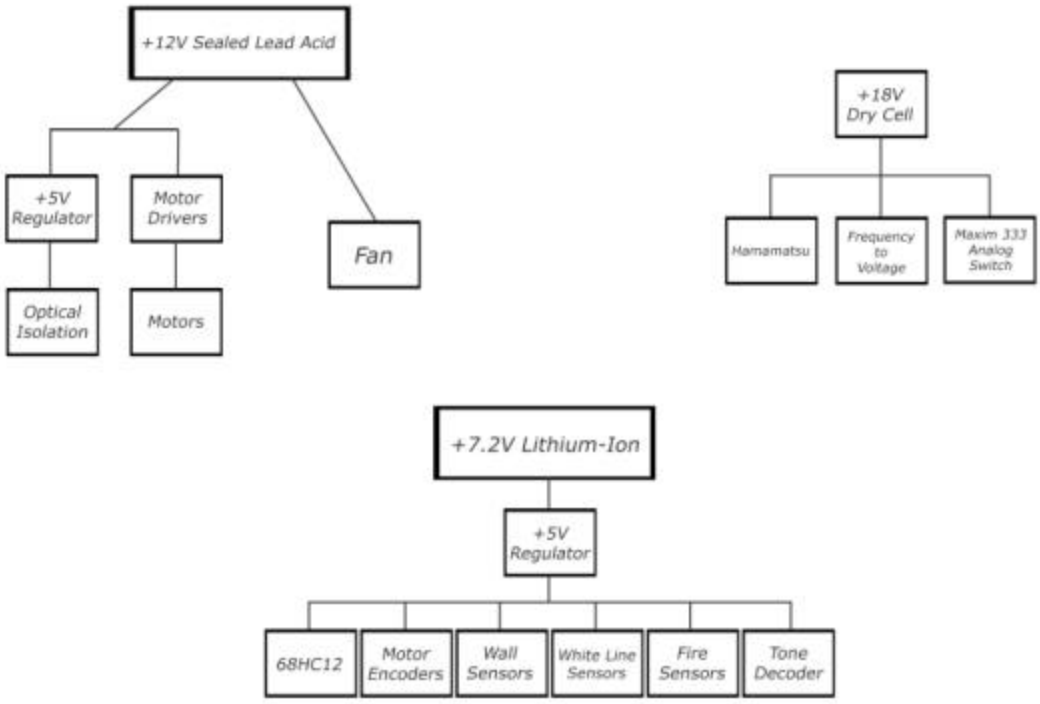



Figure #4

The Motorola 5V regulator worked flawlessly throughout the term. We did add a heat sink to it to prevent any possible overheating problems. The spec sheet states that it can provide an output current in excess of 1.0A provided that proper heat sinks are used.

The integrated circuit does have some internal thermal overload protection, but we were not about to rely on this factor in order to save the chip from possible damage.

The high power 12V battery worked remarkably well. It was able to withstand 8 hours of testing without showing any signs of fatigue. We did have a second, back-up, 7.2V battery. This was an important factor for all of our testing purposes. The length of time the battery would last was about 3.5 hours. When it would die out, we simply had to replace it with the charged back up. We were always sure to have a fully charged 7.2V battery. The 18V supply lasted longer than the other supplies. We only had to replace the 9V batteries 4 times throughout the semester.

We were satisfied with the performance of the sealed lead acid battery more than any other of the supplies. We did have to trade the cost of having such a heavy battery for the advantage of having the length of time for testing the robot. Not having the time for testing that we required would have hurt us more in the long run than going with lighter batteries.

## Switches and Fuses

Through the use of switches and fuses we were able to have security and control of all hardware components. The robot contained a total of 4 switches and 3 fuses. A switch was placed on the power to the Hamamatsu due to the fact that 18 volts powered its circuit board. A switch was connected to the motors that were powered by 12 volts. The white line sensors had a switch that powered all three of them. Finally a switch was placed on the low power side, to control the power of the cell phone battery. The benefit

to having switches on various components gave us the ability to only power what was being used at a given time. Depending on the tests we were running, we only needed to power various components. For example, if we were working on wall following, then we didn't need to power the white line sensors or the Hamamatsu. To safely secure short circuit problems we decided to fuse protect all hardware components. The motors had a 2-amp fuse, the fan had a 4-amp fuse, and the low power had a 1-amp fuse. By protecting our hardware with fuses we were able to prevent any components from being damaged due to short circuits. We did test the fuses a couple of times when wiring low power circuits. We blew a total of 3 fuses, which saved our circuitry and HC12. The use of fuses is extremely important to protect hardware; they can never hurt the robot, they simply protect circuitry.

## Wall Sensors

As the robot navigates the maze, it must have some sort of wall detection so it will not hit or touch any of the walls. The objective is to wall follow in the process of navigation. While following walls you must consider the left, front, and right walls. At the time of implementing wall sensors the only option available to us were the sharp GP2D12 IR sensors.

While testing the sensors we found that we were going to have to calibrate them. Through calibration we were going to have to determine the actual placement of them on the robot. The spec sheets indicated that the sensors were good for detecting distances of 10-80 cm. The top end of the sensitivity was great but we had to compensate for the error on the bottom end of sensing. Our ideal shortest distance to the wall was near 4 cm and the sensors were only capable of 10 cm. To correct this margin of detection we had

to calibrate them by placing them inward on the plate. By looking at the out put voltage to distance graphs, we knew we had to place them at a maximum of 10 cm inward from the plates edge. Through calibration we were actually able to correctly place them 7.5 cm from the plates edge.

Sharp GP2D12 Vo vs Distance

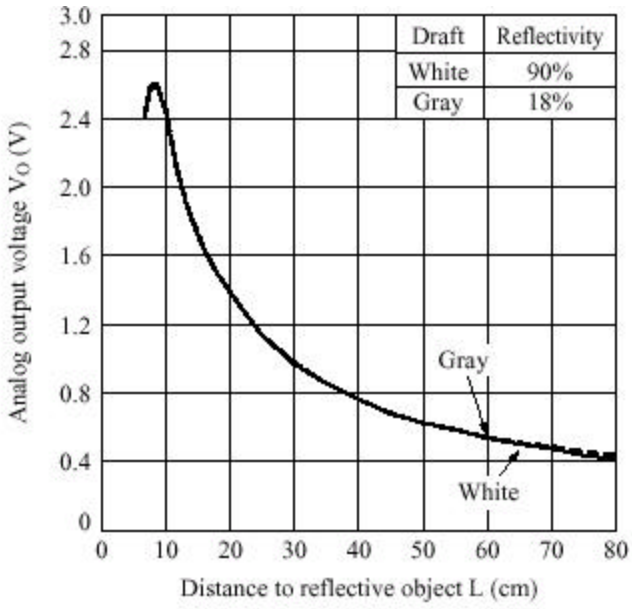



Figure #5

By placing them any less than 7.5-cm inward you then get error in detection. The error comes from the way the sensor detects the wall, as you can see from the graph. For example at 10 cm and at 8 cm you get the same out put voltage of 2.4 volts. The only way to eliminate this error is to place the sensor inwards so you can no longer detect two different distances and think they are both the same voltage.

Once the sensors were quickly calibrated, we mounted them near the front-center of the top of the plate. All sensors were placed at 7.5-cm inwards. The front sensor was placed facing in the forward direction and the two sidewall sensors were placed at an offset of 45 degrees. These sensors were pre-assembled and only required calibration. They require 5 volts for power and out put analog voltages ranging from 0-2.4 volts, with 0 volts being about 80 cm. Having to connect power, ground, and the output we were able to connect the output to an A to D port and integrate the wall sensors with the HC 12.

## White Line Sensors

The white line sensor is a very important sub system within the robot. The white line sensor can communicate with the HC 12. Through communication the robot can then understand when it has entered a room. If fire detection within the code is high, then the robot can distinguish when it has approached the white line, which is within a foot of the candle base. The white line sensor was tested and then soldered onto a perf board. Through the circuitry design the sensor is capable of adjustment, which is necessary in the event of lighting and height.

The design of the white line sensor consists of a light emitting diode (led) and a npn photo transistor both of which were powered by 5 volts. To compensate the current we had to mount some resistors within the circuit. First we placed a 50 O resistor in series on the power side of the transistor and the diode. In series with the small resister we added a 1K pot. The use of the 50 O resistor is to prevent the diode and transistor from shorting when the pot is turned down towards zero resistance. With the pot in place

the sensitivity of the sensors can then be varied to meet conditions that the robot may be faced with. These conditions can vary from ambient light to actual height from the sensor to the ground. The transistor and the diode came pre-packaged in a black plastic casing that secured both components in place. Later on we found that the LED was not putting out enough light for the transistor to signal off. We varied our pots and still could not achieve a maximum out put voltage on white. Our solution was to place a small 5-volt light on the actual perf board. With the added light we suddenly noticed a change. With some slight tune of the pots we were now able to receive about .1 volt on black and 4.85 volt on white.

White line Circuitry

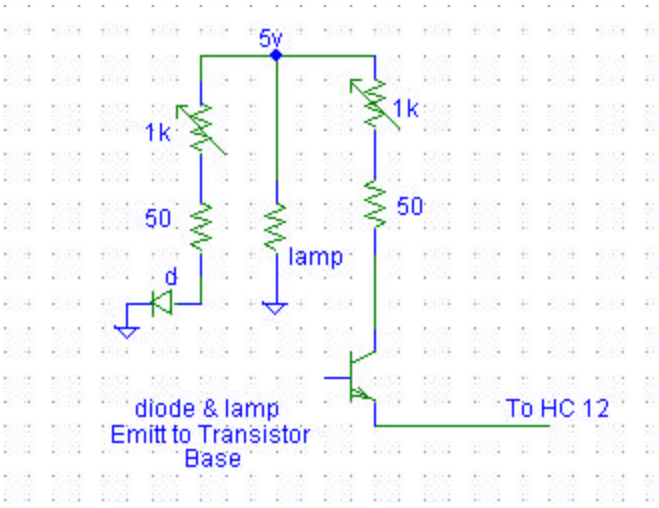



Figure #6

After designing our first white line sensor we had to make several modifications to the circuitry. With some modifications we were able to detect white lines and communicate with the HC 12. El Patron actually had three white line sensors, which

were all placed on the bottom of the main plate. The front sensor's primary function was to detect the entrance of a room and the line at the base of the candle. The other two sensors were placed at each side of the robot near the tires. These sensors were used to detect the home circle depending on the direction of the go home routine.

## Flame Sensors

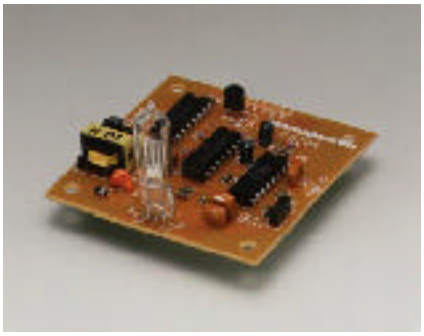
Another essential piece of El Patron was our flame sensors. We felt that this was especially important so we decided to have two flame detectors. The Primary fire sensor was a Hamamatsu ultraviolet flame detector (UVtron R2868) and its companion driving board

(C3704).

(Uvtron R2868)

Figure #7

Driving Circuit (C3704)
Figure #8



The main function of the Hamamatsu was to scan each room and determine if there was a flame present. After reading over the specification sheet we discovered that the driving board outputs pulses of different frequencies when a flame is detected. We decided to use the pulse accumulator to interface the HC12 and the Hamamatsu. We choose the Hamamatsu for its extreme sensitivity and wide range of view.

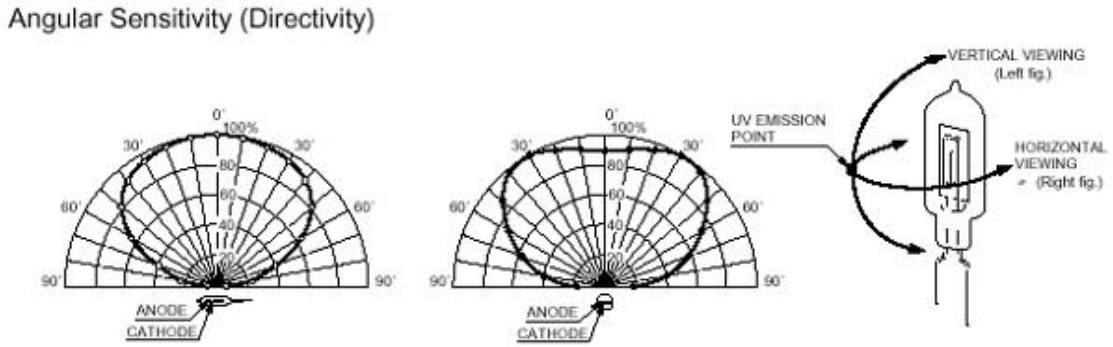



Figure #9

After several phone calls our group was able to receive a sample of this product free of charge from the Hamamatsu Corporation. Since we received this part free we about $60.00. Our secondary fire sensors were the PN168 phototransistors. The PN168's were used to home in on the flame. The PN168 were set up so that they could provide binocular vision. In order to produce binocular vision we used the casing of a pen and placed the phototransistor inside. To avoid interference from the ambient light we used the inside of a floppy disk as a filter. The floppy disk worked great in all conditions so there was no need for any adjustment. A shield was set up on the outer edge of each pen case. These shields allowed the robot to home in on a flame that may appear on the far left or right of the robot. If the flame were on the far left of the robot, only the right

sensor would pick up a reading and vice versa. A PN168 usually has a emitter voltage in the millivolt range. Because of this we felt that it would be necessary to add an amplifier to increase the voltage to 5 volts. The combination of a 10K pot and a Maxim 494 single supply operational amplifier, allowed us to increase our voltage to the desired setting and adjust the sensors to an equal value. The op-amp was used in a non-inverting configuration with the emitter of the phototransistor fed into the input of the op-amp and The 10K pot allowed us to adjust the gain of the op-amp to the desired setting. The outputs of the maxim 494 were fed into an A/D port on the HC12.

Phototransistor circuitry (PN168)

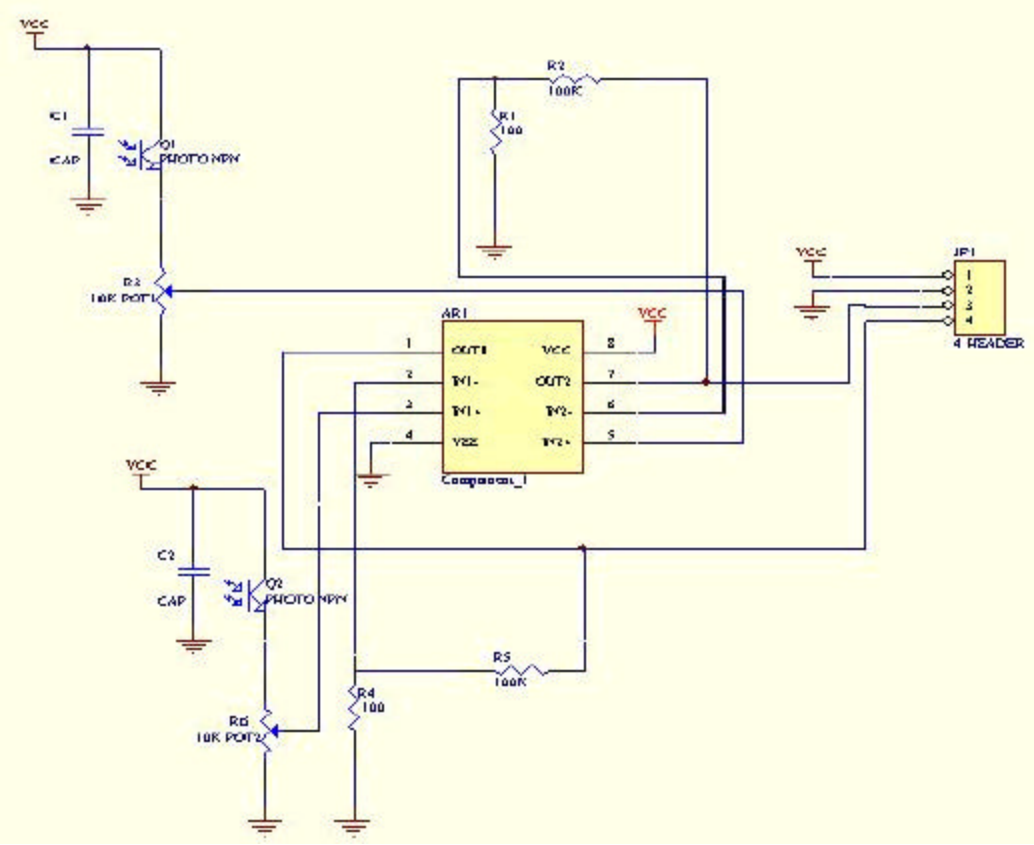



Figure #10

# Closed Loop Motor Control

To implement closed loop speed control of the motors, we used the frequency to voltage technique. The concept of this idea is trivial. Built in encoders on the motors will output a square wave at a frequency dependent on motor speed. Through frequency to voltage chips, we can transform this information to an analog voltage for the microprocessor to use to determine what speed the motors are going. The microprocessor can then adjust the speed of each motor accordingly.

Because each motor has a built in quadrature encoder, this task required two frequency to voltage chips and circuitry. We took the output signal from channel A, off of both motor encoders and fed it into the LM2917N frequency to voltage chips. Then we sent the analog voltages from the chips to the Analog to Digital Ports on the HC-12. To design the circuits, we first had to test the motors to determine their maximum output frequency. To test, we sent a 100% duty cycle to the motors from the HC-12 and measured the encoder frequency on the oscilloscope. The maximum frequency measured was 28kHz.

Frequency to Voltage Circuit

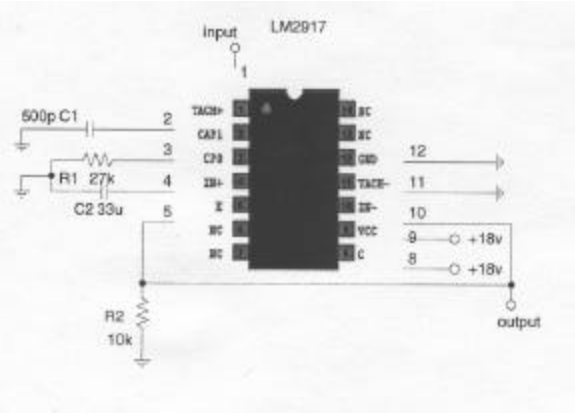



Figure #11

Using the equations in the spec sheets, we were able to create a circuit to meet our needs.

$V_o = V_{cc} \times F_{in} \times C1 \times R1$

$V_{ripple,\ p\text{-}p} = (V_{cc})/2 \times C1/C2 \times (1 - (V_{cc} \times F_{in} \times C1)/2)$

$F_{max} = I2/(C1 \times V_{cc})$

We found out that C2 determines the ripple voltage, and R1 and C1 determine the frequency to voltage range. After testing for a while, we chose C1 to be 500pf, C2 to be .33uf, and R1 to be 27k.

We powered the chips with our 18V low power supply. The reason for this was because we wanted the output to range from 0 to 5 volts. This idea ruled out the 5V source because we needed a higher voltage to correctly bias the internal transistors. We did not want to use the 12V high power supply because this would require extra isolation. Our 18V power source was great for this application. Before we made our final circuit board, we tested the circuits on breadboard. When the circuits proved to be successful, we wire wrapped and mounted the design on the robot.

## H-Bridge Motor Driver

To drive the Maxon motors, we used the dual full-bridge Allegro 2998. The single chip has dual drivers within, allowing us to run both motors from one integrated circuit. We could control each motors speed and direction directly from the HC-12 (through isolation). Each bridge operates with output currents of up to 2A continuous or 3A peak (start-up). The inputs are all TTL compatible. We had no problems sending the PWM signals or the direction signals from the HC-12.

Allegro 2998 Dual Full-Bridge Motor Driver

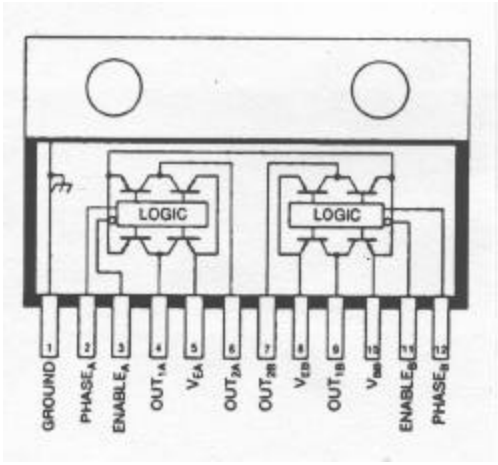



Figure #12

We powered the driver with our high power 12V supply. The chip has a voltage rating of 50V so we had no worries about overpowering it with voltage. The chip itself, however, is highly electrostatic sensitive; therefore, we had to take precautions when handling it. Also, due to its sensitivity, we prepared a second back-up identical driver circuit. In our testing, we never experienced excessive heat from this chip. We concluded that external heat sinks were not necessary for our usage.

We had very successful results from this motor driver. It had the ability to make the robot go faster than needed. We also had the ability to run the robot for hours of testing without experiencing any battery problems. In addition, the drivers also had the ability to support the starting torque requirements for larger 2.25 diameter wheels, which we tested but did not finally use.

## Optical Isolation

In order to protect the HC12 from possible surges of high voltage or current, this year the instructors decided to require some sort of isolation between the high and low power. To accomplish this, we decided to us H11L1 optical isolators from Fairchild Semi-conductors. The H11L1 series has a medium to high-speed integrated circuit detector optically coupled to a gallium-arsenide infrared emitter diode. The output incorporates a Schmitt trigger, which provides hysteresis for noise protection and pulse shaping. The detector circuit is optimized for simplicity of operation and utilizes an open collector output for maximum application flexibility. Several ports from the HC12 are connected to the h-bridge circuit. The h-bridge circuit can produce currents up to 3 Amps. High currents like these can damage the HC12 and any other circuitry that is connected. Placing the isolators between the HC12 and the h-bridge not only protects the low power from high current but also helps prevent noise interference.

Optical Isolation Circuit

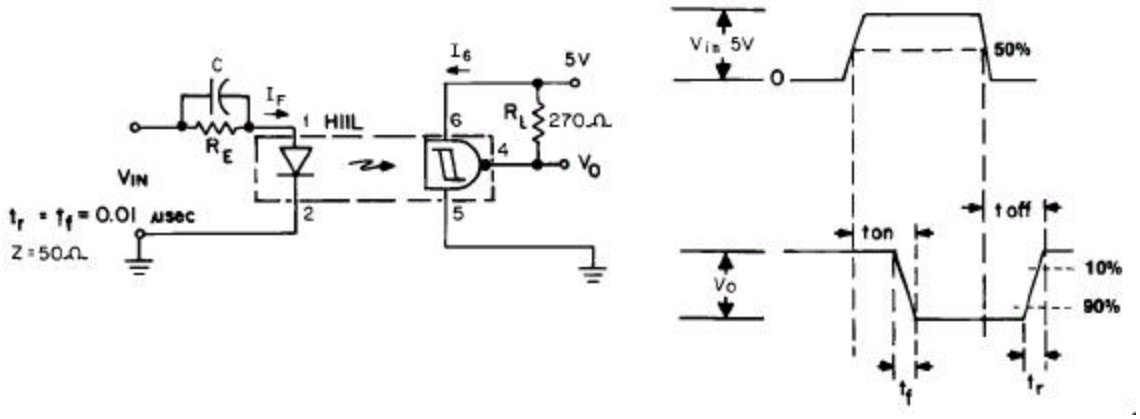



Figure #13

## Maxon DC Motor

The motors used were the Maxon, 22mm, 6W motors. The power rating, gear ratio, encoders, and size were the most appealing aspects of this motor.

The motor is rated at 6W for a nominal voltage of 18V and a starting current of 1.28A. These characteristics matched our high power supply and our Allegro motor driver specs. At a maximum charge, our sealed lead acid battery would be at 14.4V, which is within operating range. The advantage of using a 6W motor over an 11W motor is that less power is needed and consumed.

The motor is geared with a 14:1 planetary gear ratio. This would enable the robot to have enough torque to start out and still have a good deal of top end speed. We could have gone with a smaller gear ratio and had greater top end speed and less torque, or we could have gone with the inverse with a larger ratio. However, we wanted starting torque, more than speed, because we had only estimated the robot's weight in our preliminary calculations. Also, we wanted to have the ability to upgrade to larger tires if we needed to for the non-dead reckoning option. Larger tires would need more starting torque.

The built-in quadrature encoders came with a 500 count per revolution resolution. This feature was incorporated into the closed loop speed control. At the initial time of design we were concerned with having accuracy for the motor feedback portion. We could have still been quite accurate had we went with only a 100-count encoder.

The physical size of the motors was small in comparison with what we could have chosen from. They were also lightweight. We wanted to keep the overall design as light as possible, so this was a plus. Additionally, we wanted to keep the diameter of the robot as small as possible. Because the motor length was small enough, we were able to keep within a 10-inch diameter.

We had minimal problems with the 6W Maxon motor. It was easy to mount and connect to our circuitry. For the most part it was an excellent choice. The only portion that gave us trouble was in one of the encoders. Sometime in term the encoder started malfunctioning. Other than this problem, all went well.

## Tone Decoder

By implementing the tone decoder to the robot we were able to receive additional reductions on the total time El Patron took to put out the fire. This was not required by the course but we implemented it for the competition. The decoder is implemented using a LM 567 NC chip along with some circuitry for filtering. The following is a diagram of the circuitry that was etched onto a board for the decoder. By placing a pot on pin 6 we were able to control the oscillation frequency which the chip would actually latch onto. The circuit consists of an output filter and loop low pass filter. The frequency that the decoder latches onto is 3.5 KHz plus or minus 5%. The use of the pot allows some adjustment, but not much as far as detection frequency. With 5% we were able to set the frequency and never had to adjust it once it was set. The frequency signal that is sent to the decoder comes from a 9-volt buzzer. The buzzer was pre assembled and we just had to add 9 volts and a switch. With the tone decoder outputting an active low when we sent

it a 3.5 KHz signal, we were then able to communicate with the HC 12 through Port A. With the tone decoder in place, we could now have the robot activated as if a fire alarm actually set off the robot. Component values are C2 = 473uF, Rl = 1K, C3 = .83uF, C1 = 333uF, R1 = 10K, R3 = 120K, in series with R1 we placed a 10K pot for adjustment on the current controlled oscillator.

Decoder Circuit

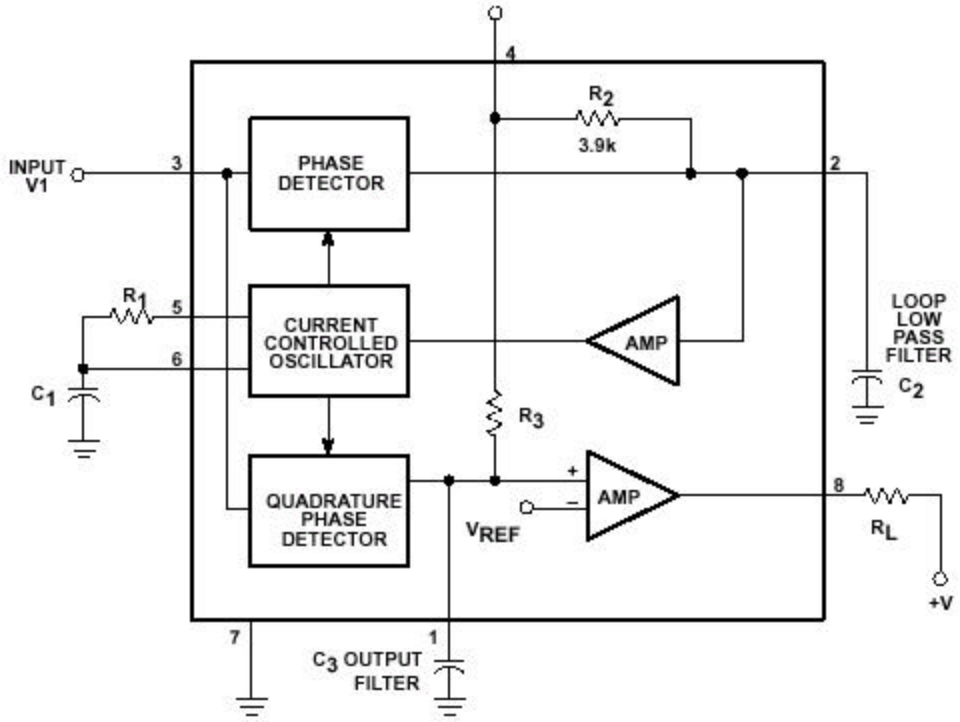



Figure #14

## Fire Extinguishing

Once El Patron had successfully reached the base of the candle and stopped at the white line, it was then time to extinguish the flame from the candle. To successfully complete this task we had to use a fan for fire extinguishing. The fan is located near the

center-front of the robot. We securely mounted the fan to an elevated plate, which, serves as a riser. The center of the fan blade is elevated to height of 17.5 cm. This elevation is the center of the flame on your average candle that is burning.
With the fan mounted it was then time to communicate with the HC 12. Through the use of a relay we were able to enable the fans power and extinguish the flame. We used an Antex Electronics ODC-01 5 volt relay. The fan is powered by the 12-volt lead acid, which gives it plenty of current. With the fan powered we communicated with the relay through an active high signal from the HC 12. Once the HC 12 sent a 5-volt high then the fan was powered until the flame was extinguished.

## HC-12 Setup

The Motorola HC-12 microprocessor gave the command of the 1st step of making the robot move to the last step of making the robot stop at home. We had to set up the expanded port A and B, timer and pulse accumulator, pulse width modulator, and (expanded) analog to digital converter subsystems.

We set up the Pulse Width Modulator to output a frequency of 1kHz. We controlled the right motor ("motor 1") through channel 1 of port p and the left motor ("motor 2") with channel 2. With this we could also control what motor speed the motors would run at. Next we set up the expanded port A (EPA) to be an output port. Channel 1 of EPA would control the direction of motor 1, and channel 2 of EPA would control the direction of motor 2.

Next, we configured the analog to digital converter to be able to scan all of our sensor values into its registers. Here we ran into a problem. There are eight inputs

(Pads) to the A/D converter. However, Pad0 and Pad1 are already taken by the HC-12 for internal reasons. This leaves 6 inputs to the A/D. For our robot design, we needed 8 analog devices that the HC-12 had to be able to read at any moment. At this point we incorporated the Maxim 333 quad CMOS analog switch.

By adding in the 333, we were able to now use up to 10 analog devices for the HC-12 to use. We then connected the front, left, and right wall sensors, as well as the front white line sensor to the processor directly. We connected the left and right flame detectors and frequency to voltage outputs to the analog switch, which was then connected to the HC-12. Now the HC-12 simply had to send a high from EPA to the 333 to tell it what information it needed. We had no problems with the switching time of the 333 and the HC-12.

Next we turned on the pulse accumulator and configured the pin as an input. We then connected the Hamamatsu to the pulse accumulator. The pulse accumulator is a 16-bit count register. Since we only needed 1 count to tell us if there was a fire present, this was a bit excessive. Nevertheless, the accumulator never let us down.

In addition to the above, we also used the output EPA to control the fan operation. The expanded port B, EPB, was configured to be an input port. It took the signals from our left and right white line sensors, as well as the signal from the tone decoder circuit. We had no problems from lack of I/O ports. If we needed any more ports, we could have easily used any one of the 5 ports from the Byte Data Link Communications Module, 4 ports from port P, or the 2 from port S.

## Program Development

After all hardware was wired up and ready to go, the next task at hand was to interface the hardware with the HC12 using C programming language. Before we started to write any code we first wrote out some pseudo code. The pseudo code allowed us to plan out what we wanted our robot to accomplish. We started out deciding if we would left or right wall follow. The reason why we choose to left wall follow is because we felt it would make it easier to enter room four. Next we needed to decide if we were going to scan each room or try to detect the flame just by passing by each room. We decided that it would be best if we would scan each room. Scanning each room would take more time but we felt it would decrease our chances of missing a flame. Once all our pseudo code was written up it was time to convert it all into the program that would navigate El Patron through the maze.

Once El Patron was placed on the home circle, he would remain there until the correct tone was heard. As soon as the tone was detected the robot would go into its left wall following routine. El Patron would continue to wall follow until its white line detector would sense a white line. The robot would then peak into the room for about three seconds. This would allow the Hamamatsu time to determine if there was a flame present. If there was no flame detected El Patron would back out, make a 90 degree turn to the left, and continue on with its left wall following. If no flames were found in rooms one through three, then El Patron would enter its room four subroutine. The room four routine would continue to left wall follow until the left wall sensor did not detect a left wall. At that point the robot would go straight until the front wall sensor picked up a wall. Once this happened the robot would make a 90-degree turn to the left. Then El Patron

would right wall follow into room four. As soon as the Hamamatsu would detect a flame, the secondary fire sensors would turn on and move towards the flame by trying to keep the voltages on the PN168s equal. Once there was a white line detected the robot would stop and extinguish the flame.

The next routine that had to be implemented was the go home. As the robot was making its way around the maze it would also count the number of white line it ran over. Counting these lines allowed our robot to know what room it detected the flame in. This number is then sent back to the go home subroutine. The go home subroutine would then determine what section of code to run in order to get home from that room. The robot would know that it was at the home circle if either the front and left white line sensor or the front and right white line sensor would detect white.

# Budget

| Item | Quantity | Unit Price | Total |
|---|---|---|---|
| Motors | 2 | $50.00 | $100.00 |
| Battery | 1 | $30.00 | $30.00 |
| Charger | 1 | $9.00 | $9.00 |
| Charger Plug | 1 | $2.00 | $2.00 |
| Fuse Holder | 2 | $1.00 | $2.00 |
| 3-way Switch | 1 | $0.50 | $0.50 |
| 2-way Switch | 1 | $0.50 | $0.50 |
| Fan Relay Switch | 2 | $5.00 | $10.00 |
| Caster Wheel | 1 | $10.00 | $10.00 |
| 2A Fuses | 2 | $0.20 | $0.40 |
| Allegro H Bridge | 2 | $7.00 | $14.00 |
| Optical Isolators | 4 | $1.50 | $6.00 |
| 6 Pin Sockets | 8 | $0.20 | $1.60 |
| 5V Regulator | 3 | $0.75 | $2.25 |
| .5A Fuse | 1 | $1.00 | $1.00 |
| 2 pin male connector | 2 | $0.75 | $1.50 |
| 2 pin female connector | 2 | $0.75 | $1.50 |
| 4 pin male connector | 2 | $1.25 | $2.50 |
| 4 pin female connector | 2 | " | |
| 6 pin male connector | 2 | $2.50 | $5.00 |
| 6 pin female connector | 2 | " | |
| Plug Connectors | 12 | " | |
| Motor Connector Pins | 4 | $0.10 | $0.40 |
| Wall Sensors GP2D-12 | 3 | $10.00 | $30.00 |
| Wall Sensor Plugs | 3 | " | |
| White Line Sensor | 3 | $1.92 | $5.76 |
| Perfboard | 1 | $1.00 | $1.00 |
| 1/2 in. Round Spacers | 14 | $0.25 | $3.50 |
| Led Lights | 3 | $1.00 | $3.00 |
| 1k pots | 6 | $0.50 | $3.00 |
| Prescription bottle | 1 | $0.10 | $0.10 |
| small rubber clamps | 2 | $0.50 | $1.00 |
| 4-pin connectors | 3 | $0.60 | $1.80 |
| 9V buzzer | 1 | $1.50 | $1.50 |
| **TOTAL** | | | $250.81 |

FREE ITEMS

Memory Expansion
Fan and Motor
Wheels and Rims
Metal Plate
Heat Sink
1-W Resistors
Hamamatsu
9-Volt Batteries
7V Battery & Charger
Tone Decoder
Maxim 333
Maxim 494
24K Gold NM Tech Flag
Nuts & Bolts from Norton

## Conclusion

Our goal at the beginning of the semester was to build a fire-fighting robot. We had to demonstrate all of the skills that we have acquired throughout our electrical engineering background. By working together as a team of three we had to learn how to become productive by communicating among each other. By allowing each group member to excel in their areas of electrical engineering we were all able to successfully complete programming and sub systems within El Patron. Through excellent communication we were also able to collaborate and help each other in each individuals area of interest and specialty. With hard work and dedication we were able to successfully build a fire-fighting robot El Patron. We qualified to be in the national competition at Trinity College. Despite some problems, we finished 21$^{st}$ out of 71 entries. By correcting some of the problems, we finished 3$^{rd}$ at the local competition. As a group we completed our task and feel that we all put in 100% into building El Patron at a total cost of $254.00.

# CODE

```
#include "hc12.h"
#include "DBug12.h"
#define TRUE  1
#define FALSE 0
#define PORTEA (* (unsigned char *) (0x400))
#define PORTEB (* (unsigned char *) (0x401))
#define DDREAB (* (unsigned char *) (0x402))

void delay(int num);
void home(int T);

volatile unsigned int first, second, time, flameA, flameB, w, e, room ;
volatile unsigned int whiteline, wl, cline, x, dline, f_count, a, b, hcnt ;
unsigned char VOLT, FRONT, LINE, VOLTR;

main()
{

        DDREAB = 0X01;                          /*  portb=input  porta= output*/

        PWCLK = PWCLK  & ~0xC0;                 /*   8-Bit mode                */
        PWPOL = PWPOL  |  0x0F;                 /*   HIGH POLARITY             */
        PWCTL = PWCTL  & ~0x04;                 /*    Left alligned            */
        PWPOL = PWPOL & ~0x80;                  /*   clock mode B              */
        PWCLK =  (PWCLK | 0x04)  & ~0x03;       /* set n=4                     */
        PWPER1 = 249;                           /* SELECT PERIOD  ch1          */
        PWPER2 = 249;                           /* SELECT PERIOD ch2           */
        PWEN   =  PWEN | 0x06;                  /*enable ch1 and ch2           */

         TSCR = 0x80;                           /* Turn on timer subsystem */

        DDRT = 0x7F;
        PACTL = 0X50;
        PACNT = 0000;

        /* Turn on timer subsystem */
        TSCR = 0x80;
        /* Set prescaler to 32 */
        TMSK2 = 0x05;

        ATDCTL2 =  0X80;                /* POWER UP THE A/D CONVERTER */
        ATDCTL4 = 0X01;                 /* SET THE CLK SOURCE  */
        ATDCTL5 = 0X72;                 /* SCAN=1, MULT=1 */
```

```
                room = 0;
                w=0;
                e=0;
                x=1;    /*  room 3 clear  */
                dline=0;
                cline=0;
                f_count=0;
                a=0;
                b=0;
                hcnt=0;
while ( (PORTEB == 0x01) | (PORTEB == 0x03) | (PORTEB == 0x07) | (PORTEB ==0x05)  )  /* tone start */
                {
                 LINE = ADR7H;       /* white line sensor */
                 PWDTY1=0;
                 PWDTY2=0;
                 PACNT = 0000;
                }

                PORTEA = 0x00;
                PWDTY1 = 220;
                PWDTY2 = 240;
                delay (1000);

                while (1)
                {
                        /*  START SEARCHING ROOMS AND ROOM COUNTING  */

                  VOLT = ADR2H;    /* SENSOR VOLTAGE IN ON PAD2 */
                  VOLTR = ADR3H;
                  FRONT = ADR4H;  /* Front SENSOR PAD3 */
                  LINE = ADR7H;      /* white line sensor */
                  flameA = ADR5H;   /* RIGHT FIRE SENSOR VOLTAGE*/
                  flameB = ADR6H;   /* LEFT FIRE SENSOR VOLTAGE*/
                  PORTEA = 0X00;


                        if  (VOLT > 0X48) /*  speed B up  turn away from wall*/
                        {
                         PORTEA = 0x00;
                         PWDTY1 = 40;
                         PWDTY2 = 80;
                        }

                        else if (VOLT < 0X48) /* slow B dn   turn toward wall  */
                        {
                         PORTEA = 0x00;
```

```
        PWDTY2 = 40 ;
        PWDTY1 = 80;

                if (VOLT < 0x35)   /*25,35  sharp left turn   */
                {
                  PWDTY2 = 29;
                  PWDTY1 = 157;
                }
      }

      if (FRONT > 0x45)  /*  Head on ...turn right..sharp  */
      {
        PORTEA = 0x02;
        PWDTY2 = 220 ;
        PWDTY1 = 220;
        delay (450);
      }

      if (LINE > 0xD0)  /* line routine */
      {
       w = 1;
       PWDTY2 = 0;
       PWDTY1 = 0;
delay (1000);
       LINE = ADR7H;       /* white line sensor */

                if (( w !=0 ) && ( LINE < 0xD0 ))   /*  room counter  */
                {
                 room = room + 1;
                 w = 0;
                }

                while(PACNT > 0x0000)   /* check Hama...fire present!!!!!!!!  */
                {
                  LINE = ADR7H;      /* white line sensor */
                  flameA = ADR5H;   /* RIGHT FIRE SENSOR VOLTAGE*/
                  flameB = ADR6H;   /* LEFT FIRE SENSOR VOLTAGE*/
                 VOLT = ADR2H;     /* SENSOR VOLTAGE IN ON PAD2 */
                 VOLTR = ADR3H;
                 FRONT = ADR4H;  /* Front SENSOR PAD3 */
                 PORTEA = PORTEA | 0x30;

                        if (w==0)  /*  jump fwd 2 avoid doorway  */
                        {
                        PORTEA = 0x00;
                        PWDTY1 = 100;        /* 150 230 */
```

```
        PWDTY2 = 100;
        delay (500);
        w=1;
        }

if (flameA > flameB)
        {
         PWDTY1 = 80;        /*250*/
         PWDTY2 = 40;        /*50*/
        }

        else if(flameA < flameB)
        {
         PWDTY1 = 40;
         PWDTY2 = 80;
        }

        if  (VOLT > 0X4A) /*  speed B up  turn away from wall*/
        {
         PORTEA = 0x30;
         PWDTY1 = 40;
         PWDTY2 = 190; /*120*/
        }

        if  (VOLTR > 0X4A)
        {
         PORTEA = 0x30;
         PWDTY2 = 40;
         PWDTY1 = 190; /*120*/
        }

        if (FRONT > 0x48)  /*  Head on ...right turn  */
        {
         PORTEA = 0x32;
         PWDTY2 = 220 ;
         PWDTY1 = 220;
         delay (300);
        }

        while ( LINE > 0xE0)
        {
         PORTEA = 0x00;
         PWDTY1 = 0;
         PWDTY2 = 0;

         PORTEA = 0x01;
```

```
                        delay(800);
                        PORTEA = 0x00;
                        PWDTY1 = 20;
                        PWDTY2 = 20;
                        delay(400);

                              while(1)
                              {
                               PWDTY1 = 0;
                               PWDTY2 = 0;
                               home(room);
                              }
                      }
                } /*end HAMA check*/

/*  GO AND CHECK ANOTHER ROOM IN THE MAZE  */
                        PORTEA = 0x06;  /*   if no fire reverse to line routine */
                        PWDTY1 = 90;
                        PWDTY2 = 90;
                        delay (650);    /*  reverse  */
                        PORTEA = 0x02;      /* 90-degree cw turn */
                        PWDTY2 = 150 ;
                        PWDTY1 = 150;
                        delay (420);    /*  rotation pause */
                        PORTEA = 0x00;
                        PWDTY2 = 100 ;
                        PWDTY1 = 100;
                        delay (450);    /* speed up to detect next left wall*/

                        if ((room==3) && (PACNT==0000) && (x==1))
                        {
                         x=0;
                        }
              }  /*  end the line count and clear routine  */

/*  GO AND CHECK ROOM 4 */
                while (x==0)
                {

                        LINE = ADR7H;       /* white line sensor */
                        flameA = ADR5H;     /* RIGHT FIRE SENSOR VOLTAGE*/
                        flameB = ADR6H;     /* LEFT FIRE SENSOR VOLTAGE*/
                        VOLT = ADR2H;       /* SENSOR VOLTAGE IN ON PAD2 */
                        VOLTR = ADR3H;
                        FRONT = ADR4H;  /* Front SENSOR PAD3 */
```

```
   room=4;

   if  (VOLT > 0X48) /*  speed B up  turn away from wall*/
   {
   PORTEA = 0x00;
   PWDTY1 = 40;
   PWDTY2 = 80;
   }

   else if (VOLT < 0X48) /* slow B dn   turn toward wall  */
   {
   PORTEA = 0x00;
   PWDTY2 = 40 ;
   PWDTY1 = 80;
}

   if (FRONT > 0x45)  /*  Head on ...turn right..sharp  */
   {
    PORTEA = 0x02;
    PWDTY2 = 220 ;
    PWDTY1 = 220;
    delay (180);
   }

   if (VOLT < 0x35)   /*looking to ...SEE TURN AT LEFT   */
   {
            while (1)
            {
              LINE = ADR7H;      /* white line sensor */
              flameA = ADR5H;   /* RIGHT FIRE SENSOR*/
              flameB = ADR6H;   /* LEFT FIRE SENSOR */
              VOLT = ADR2H;    /* L SENSOR VOLTAGE  */
              VOLTR = ADR3H;
              FRONT = ADR4H;  /* Front SENSOR PAD3 */

   while (FRONT < 0x47)  /*  drive strait till front wall is close  */
                        {
                         PORTEA = 0x00;
                         PWDTY1 = 110;
                         PWDTY2 = 80;
                         FRONT = ADR4H;  /* Front SENSOR PAD3 */
                        }

                        if (FRONT > 0x48)
                        {
```

```
while (1)
{
  LINE = ADR7H;      /* white line sensor */
  flameA = ADR5H;   /* RIGHT FIRE SENSOR VOLTAGE*/
  flameB = ADR6H;   /* LEFT FIRE SENSOR VOLTAGE*/
  VOLT = ADR2H;    /* SENSOR VOLTAGE IN ON PAD2 */
  VOLTR = ADR3H;
   FRONT = ADR4H;  /* Front SENSOR PAD3 */


        if (FRONT > 0x45)  /*  Head on ...turn left..sharp  */
                                {
                                 PORTEA = 0x04;
                                 PWDTY2 = 150 ;
                                 PWDTY1 = 150;
                                 delay (420);
                                }

        if  (VOLTR > 0X43) /*  away from wall*/
                                {
                                 PORTEA = 0x00;
                                 PWDTY1 = 90;
                                 PWDTY2 = 45;
                                }

        else if (VOLTR < 0X43) /* slow B dn   turn toward wall  */
                                {
                                 PORTEA = 0x00;
                                 PWDTY2 = 90 ;
                                 PWDTY1 = 45;

        if (VOLTR < 0x35)   /*sharp right turn into room 4  */
                                        {
                                         PWDTY2 = 160;
                                         PWDTY1 = 42;
                                        }
                                }

                                if (LINE > 0xD0)  /* line routine */
                                {
                                 w = 1;
                                 PWDTY2 = 0;
                                 PWDTY1 = 0;
                                 delay (1000);
                                 LINE = ADR7H;
```

```
                                        if (( w !=0 ) && ( LINE < 0xD0 ))
/*  room counter  */
                                        {
                                         f_count = f_count + 1;
                                         w = 0;
                                        }
/*  check HAMA now at room 4 */
                              while(f_count==1)     /*  PACNT > 0x0000)     */
                              {
                              LINE = ADR7H;        /* white line sensor */
                              flameA = ADR5H;     /* RIGHT FIRE SENSOR */
                              flameB = ADR6H;
                              VOLT = ADR2H;
                              VOLTR = ADR3H;
                               FRONT = ADR4H;
                               PORTEA = PORTEA | 0x30;

                              if (w==0)  /*  jump fwd 2 avoid doorway  */
                                                {
                                                PORTEA = 0x00;
                                                PWDTY1 = 100;
                                                PWDTY2 = 100;
                                                delay (200);
                                                w=1;
                                                }

                                        if (flameA > flameB)
                                                {
                                                PWDTY1 = 80;
                                                PWDTY2 = 40;
                                                }

                                                else if(flameA < flameB)
                                                {
                                                PWDTY1 = 40;
                                                PWDTY2 = 80;
                                                }

                                                if  (VOLT > 0X4A)
                                                {
                                                PORTEA = 0x30;
                                                PWDTY1 = 40;
                                                PWDTY2 = 190;
                                                }
```

```
                                                    if  (VOLTR > 0X4A)
                                                    {
                                                    PORTEA = 0x30;
                                                    PWDTY2 = 40;
                                                    PWDTY1 = 190;
                                                    }

                                                while ( (LINE > 0xE0) && (f_count==1) )
                                                {
                                                 PWDTY1 = 0;
                                                 PWDTY2 = 0;
                                                 PORTEA = 0x01;
                                                 delay(800);
                                                 PORTEA = 0x00;
                                                 PWDTY1 = 20;
                                                 PWDTY2 = 20;
                                                 delay(400);

                                                while(1)
                                                {
                                                  PWDTY1 = 0;
                                                  PWDTY2 = 0;
                                                  home(room);
                                                }  /*ends go home loop escape  */
                                             }
                                           }  /*end HAMA check*/
                                    }                                       }
                                }
                            }
                        }
                }

                }
} /*end main*/
```

```
void delay(int num)
{
   int i;

   while (num>0)       /* Out loop delays num ms */
   {
      i = 1333;       /* Inner loop takes 6 cycles */
      while (i > 0)   /* 1333 times 6 = 1 ms */
      {
         i = i-1;
      }
      num = num - 1;
   }
}
```

```
void home(int T)        /*  GO HOME FROM ANY ROOM ROUTINE    */
{
 whiteline = T;
 cline = 0;
 hcnt = 0;
                  if(whiteline==1)   /*  GO HOME FROM ROOM 1   */
                           {

                                    VOLT = ADR2H;
                                    VOLTR = ADR3H;
                                    FRONT = ADR4H;
                                    PORTEA = 0x00;

                                   while (FRONT < 0x48)
                                   {
                                     PORTEA = 0x00;
                                     PWDTY1 = 80;
                                     PWDTY2 = 80;
                                     FRONT = ADR4H;  /* Front SENSOR PAD3 */
                                   }

                                   while(hcnt <= 550)
                                   {
                                            VOLT = ADR2H;
                                            VOLTR = ADR3H;
```

```
                FRONT = ADR4H;
                LINE = ADR7H;


               if  (VOLTR > 0X48)
               {
                 PORTEA = 0x00;
                 PWDTY1 = 160;
                 PWDTY2 = 40;
                }

               else if (VOLTR < 0X48)
                {
                        PORTEA = 0x00;
                        PWDTY2 = 160 ;    //245
                        PWDTY1 = 44;         //50

                       if (VOLTR < 0x35)   /*  sharp right turn   */
                       {
                        PWDTY2 = 157;
                        PWDTY1 = 27;
                       }
                }

                if (FRONT > 0x42)  /*  Head on .turn left..sharp  */
               {
                PORTEA = 0x04;
                PWDTY2 = 190 ;
                PWDTY1 = 190;
                delay (420);
               }

               hcnt = hcnt + 1;

        }

while(hcnt > 550)
        {
                VOLT = ADR2H;
                VOLTR = ADR3H;
                FRONT = ADR4H;
                LINE = ADR7H;


               if (LINE > 0xE0)
               {
```

```
                wl = 1;
                LINE = ADR7H;       /* white line sensor */
                }

            if ( ( wl !=0 ) && ( LINE < 0xE0 ) )   /*  room counter  */
                {
                cline = 5;
                wl = 0;
                }

                if  (VOLTR > 0X48)
                {
                PORTEA = 0x00;
                PWDTY1 = 160;
                PWDTY2 = 40;
                }

                else if (VOLTR < 0X48)
                {
                PORTEA = 0x00;
                PWDTY2 = 160 ;
                PWDTY1 = 44;

                        if (VOLTR < 0x35)   /*  sharp right turn   */
                        {
                        PWDTY2 = 157;
                        PWDTY1 = 27;
                        }
                }

                if (FRONT > 0x42)
                {
                PORTEA = 0x04;
                PWDTY2 = 190 ;
                PWDTY1 = 190;
                delay (420);
            }
if (((PORTEB == 0x03) && (LINE > 0xE0)) | ((PORTEB == 0x05) && (LINE > 0xE0))) /* STOP@HOME  */
                {
                while (1)
                {
                PWDTY2 = 0;
                PWDTY1 = 0;
                }

                }
```

```
        }
    }

else if(whiteline==2)   /*  GO HOME FROM ROOM 2  */
    {
        unsigned char VOLT, FRONT, LINE, VOLTR;
        VOLT = ADR2H;
        VOLTR = ADR3H;
        FRONT = ADR4H;  /* Front SENSOR PAD3 */

       while (FRONT < 0x48)
       {
         PORTEA = 0x00;
         PWDTY1 = 80;
         PWDTY2 = 80;
         FRONT = ADR4H;
       }

     while(hcnt <= 750)
       {
         VOLT = ADR2H;
         VOLTR = ADR3H;
         FRONT = ADR4H;  /* Front SENSOR PAD3 */
         LINE = ADR7H;     /* white line sensor */

       if  (VOLT > 0X48) /*  speed A up  turn away from wall*/
        {
          PORTEA = 0x00;
          PWDTY2 = 160;
          PWDTY1 = 40;
        }

       else if (VOLT < 0X48) /* slow A dn   turn toward wall  */
        {
         PORTEA = 0x00;
         PWDTY1 = 160 ;    //245
         PWDTY2 = 44;       //50

                    if (VOLT < 0x35)
                    {
                     PWDTY1 = 157;
                     PWDTY2 = 27;
                    }
                    if (VOLT < 0x08)
                    {
                     a=1;
```

```
                }
            }

            if (FRONT > 0x42)
            {
             PORTEA = 0x02;
             PWDTY2 = 190 ;
             PWDTY1 = 190;
             delay (420);
            }

            hcnt = hcnt + 1;
        }

while(hcnt > 750)
        {
            volatile unsigned int count;
            unsigned char VOLT, FRONT, LINE, VOLTR;
            VOLT = ADR2H;
            VOLTR = ADR3H;
            FRONT = ADR4H;
            LINE = ADR7H;

        if (LINE > 0xE0)
        {
            wl = 1;
            LINE = ADR7H;
        }

        if ( ( wl !=0 ) && ( LINE < 0xE0 )  )
        {
          cline = 5;
          wl = 0;
        }

        if  (VOLT > 0X46) /*  speed A up  turn away from wall*/
         {
           PORTEA = 0x00;
           PWDTY2 = 90;
           PWDTY1 = 30;
         }

        else if (VOLT < 0X46) /* slow A dn   turn toward wall  */
         {
            PORTEA = 0x00;
            PWDTY1 = 90 ;    //245
```

```
                                        PWDTY2 = 30;          //50

                                        if (VOLT < 0x40)   /* 35  sharp left turn   */
                                        {
                                         PWDTY1 = 157;
                                         PWDTY2 = 27;
                                        }
                                        if (VOLT < 0x08)
                                        {
                                         a=1;
                                        }
                                    }

                                   if (  (FRONT > 0x48) && (a==0) )
                                   {
                                    PORTEA = 0x02;
                                    PWDTY1 = 190 ;
                                    PWDTY2 = 190;
                                    delay (375);
                                   }

                                  if (  (FRONT > 0x48) && (a==1) )  /* 48 bust a bitch  */
                                   {
                                    PORTEA = 0x02;
                                    PWDTY1 = 190 ;
                                    PWDTY2 = 190;
                                    delay (725);

                                    PORTEA = 0x00;
                                    PWDTY1 = 100 ;
                                    PWDTY2 = 100;      /* 110 */
                                    delay (500);
                                  }
if (((PORTEB == 0x05) && (LINE > 0xE0)) | ((PORTEB == 0x03) && (LINE > 0xE0))) /*  STOP @ HOME  */
                                   {
                                        while (1)
                                        {
                                        PWDTY2 = 0;
                                        PWDTY1 = 0;
                                        }
                                   }
                               }
                          }


                  else if(whiteline==3)  /*  GO HOME FROM ROOM 3 ROUTINE*/
```

```
{
        VOLT = ADR2H;
        VOLTR = ADR3H;
        FRONT = ADR4H;

  while(hcnt <= 30000)
        {
                VOLT = ADR2H;
                VOLTR = ADR3H;
                FRONT = ADR4H;  /* Front SENSOR PAD3 */
                LINE = ADR7H;       /* white line sensor */

               if  (VOLT > 0X48)
                {
                  PORTEA = 0x00;
                  PWDTY2 = 160;
                  PWDTY1 = 40;
                }

               else if (VOLT < 0X48)
                {
                        PORTEA = 0x00;
                        PWDTY1 = 160 ;
                        PWDTY2 = 44;

                       if (VOLT < 0x35)
                       {
                         PWDTY1 = 157;
                         PWDTY2 = 27;
                       }
                       if (VOLT < 0x08)
                       {
                         a=1;
                       }
                }

                if (FRONT > 0x42)
               {
                 PORTEA = 0x02;
                 PWDTY2 = 190 ;
                 PWDTY1 = 190;
                 delay (420);
               }

               hcnt = hcnt + 1;
```

```
        }

while(hcnt > 30000)
        {
                VOLT = ADR2H;
                VOLTR = ADR3H;
                FRONT = ADR4H;
                LINE = ADR7H;

        if (LINE > 0xC0)
        {
                wl = 1;
                LINE = ADR7H;        /* white line sensor */
        }

        if ( ( wl !=0 ) && ( LINE < 0xC0 ) )   /*  room counter  */
        {
          cline = 5;
          wl = 0;
        }

        if  (VOLT > 0X48)
         {
           PORTEA = 0x00;
           PWDTY2 = 120;
           PWDTY1 = 40;
         }

        else if (VOLT < 0X48)
         {
                PORTEA = 0x00;
                PWDTY1 = 120 ;    //245
                PWDTY2 = 44;         //50

                if (VOLT < 0x35)
                {
                  PWDTY1 = 157;
                  PWDTY2 = 27;
                }
         }

    if (FRONT > 0x42)
        {
          PORTEA = 0x02;
          PWDTY1 = 190 ;
          PWDTY2 = 190;
```

```
                    delay (420);

                  }

                   if ( (PORTEB == 0x07) && (LINE > 0xE0) )
                   {
                          while (1)
                          {
                           PWDTY2 = 0;
                           PWDTY1 = 0;
                          }
                   }

            }
      }

else if(whiteline==4)  /*  GO HOME FROM ROOM 4*/
        {
                  while(hcnt <= 750)
                  {
                          VOLT = ADR2H;
                          VOLTR = ADR3H;
                          FRONT = ADR4H;  /* Front SENSOR PAD3 */
                          LINE = ADR7H;       /* white line sensor */

                         if  (VOLTR > 0X48)
                          {
                            PORTEA = 0x00;
                            PWDTY1 = 160;
                            PWDTY2 = 40;
                          }

                         else if (VOLTR < 0X48)
                          {
                                  PORTEA = 0x00;
                                  PWDTY2 = 160 ;    //245
                                  PWDTY1 = 44;         //50

                                 if (VOLTR < 0x35)   /*  sharp right turn   */
                                 {
                                   PWDTY2 = 157;
                           PWDTY1 = 27;
                                 }
                          }

                          if (FRONT > 0x42)
```

```
        {
         PORTEA = 0x04;
         PWDTY2 = 190 ;
         PWDTY1 = 190;
         delay (420);
}

        hcnt = hcnt + 1;
}

while(hcnt > 750)
{


         VOLT = ADR2H;
         VOLTR = ADR3H;
         FRONT = ADR4H;  /* Front SENSOR PAD3 */
         LINE = ADR7H;       /* white line sensor */

        if  (VOLTR > 0X48)
         {
         PORTEA = 0x00;
         PWDTY1 = 150;
         PWDTY2 = 40;
         }
        else if (VOLTR < 0X48)
         {
         PORTEA = 0x00;
         PWDTY2 = 120 ;
         PWDTY1 = 44;

                if (VOLTR < 0x15)   /*  sharp right turn   */
                {
                  PWDTY2 = 157;
         PWDTY1 = 27;
                }
         }

        if (FRONT > 0x42)  /*  Head on turn left..sharp  */
        {
         PORTEA = 0x04;
         PWDTY2 = 190 ;
         PWDTY1 = 190;
         delay (420);
        }
```

```
if (((PORTEB == 0x05) && (LINE > 0xE0))  | ((PORTEB == 0x03) && (LINE > 0xE0)))  /* STOP @ HOME  */

                                                {
                                                        while (1)
                                                        {
                                                        PWDTY2 = 0;
                                                        PWDTY1 = 0;
                                                        }
                                                }

                                        }
                                }

                        else
                                {
                                        VOLT = ADR2H;
                                        VOLTR = ADR3H;
                                        FRONT = ADR4H;  /* Front SENSOR PAD3 */
                                        PORTEA = 0x02;      /* 180-degree ccw turn */
                                        PWDTY2 = 220 ;
                                        PWDTY1 = 220;
                                        delay (1000);  /*  rotation pause */
                                        PORTEA = 0x00;
                                }
        }
```